下载APP

30｜怎么快速把任意文件格式转成PDF，并批量加水印？

2021-04-20 尹会生

Python自动化办公实战课 进入课程 >

讲述：尹会生
时长 18:11 大小 16.67M

你好，我是尹会生。

在办公场景中，我们打交道最多的软件，要数 Office 的办公套件了，功能丰富且强大，使用方便。不过你可能也会发现，我们经常用于文字编辑的 Word 软件，它使用的 docx 扩展名的文件无论是在不同操作系统平台、不同的设备上，还是在内容安全性和格式的兼容性上，都没有 PDF 文件强大。Excel 和 PowerPoint 中的文件也是如此。

例如：你需要把公司的合同范本发给其他公司审阅，为了保证文本不会被随意篡改，往往需要把 Word 或 PowerPoint 的文件转换为 PDF 文件，再通过邮件发给其他公司。而且，随着数字化转型的开始，原本在计算机上正常显示的表格，拿到手机上可能会缺少字体；也可能因为屏幕宽度不同，导致格式无法对齐；还可能会出现无法显示文字等格式问题。

不过这些问题呢，PDF 统统可以解决。所以像是商业的条款、合同、产品介绍，既要保证安全，又要确保格式正确，如果你不想限制用户必须要用多宽的显示器，或者必须安装好特定的字体，才能看到你的 Word、Excel、PowerPoint 里的内容的话，那么使用 PDF 文件就是非常必要的了。

所以在今天这一讲中，我将带你学习如何把 Word、Excel、PowerPoint 的默认文件格式批量转换为 PDF 文件，并教你怎么给 PDF 文件增加水印，既保证样式美观，又确保文档的安全。

将常用办公文件转换为 PDF 格式

如果你以前遇到过类似转换为 PDF 文件的需求，那么在你搜索 Python 的第三方库时，就会发现，将 Word、Excel、PowerPoint 的默认文件保存格式转换为 PDF 的库非常多。由于我要将多种格式的文件转换为 PDF，那么为了我就需要使用一个能支持 Office 所有组件的库，而这个库就是 pywin32 库。

pywin32 库支持了绝大多数的 Windows API，因此它只能运行在操作系统是 Windows 上，当你需要使用 pywin32 操作 Office 各个组件时，可以利用 pywin32 库调用 Offcie 组件中的 VBA 实现 Office 组件的大部分功能。

接下来，我将从 pywin32 库的安装开始，来为你逐步讲解一下如何把 Office 组件中的 Word、Excel、PowerPoint 的默认文件保存格式转换为 PDF 文件。

虽然你要学习三种不同的软件格式转换为 PDF 文件，但是它们三个文件格式转换的思路和被调用的函数是相同的，你可以通过对照 Word 文件转换为 PDF 来去掌握其他两个软件的文件格式转换代码，来学习自动格式转换，这样学起来会更加轻松。

将 Word 文档转换为 PDF

我先以 Word 为例，来为你讲解一下 pywin32 库的安装、导入，以及将 Word 文档进行转换的操作步骤。

由于 pywin32 是第三方库，所以你要使用 pip 命令把它安装到你的 Windows 计算机中。这里需要注意，pywin32 的安装包和软件名称不同，而且导入的库名称也和 pywin32 不同。所以我把 pywin32 库的安装和导入时，使用的名称写在文档中，供你进行参考：

复制代码

```
SHELL$ pip3 install pypiwin32
PYTHON> import win32com
```

我们**用于格式转换的库**叫做**“win32com”**，它是 pywin32 软件的库名称。安装它时，你要使用 pypiwin32，作为它的安装包名称来使用。这是第一次接触该库最容易混淆的地方，我建议你在阅读后续内容前，先对“pywin32、pypiwin32、win32com”这三个概念进行区分。

在你正确区分上面三个概念之后，我们就可以开始导入 win32com 库，并调用 VBA 脚本，来把 Word 文档转换为 PDF 格式了。

为了让你更好地理解 win32com 库的执行过程，我来为你举个例子。我在"C:\data"文件夹下有一个 Word 格式的 a.doc 文件，现在要将它自动转换为 a.pdf 文件，并继续存储在该目录中。如果你手动进行格式转换，只是需要以下四个步骤：

1. 进入到 C:\data 文件夹；
2. 使用 Office 的 Word 组件打开 a.doc 文件；
3. 使用“文件另存为”功能，保存为 PDF 格式，并指定保存目录；
4. 保存并关闭 Word 文件，退出 Word 进程。

由于 win32com 库是调用 Word 的 VBA 脚本实现的格式转换功能，因此转换格式的 Python 代码步骤也和手动操作 Word 几乎相同，少数不同的地方是因为 win32com 支持的组件较多，需要指定当前转换采用的 VBA 脚本为 Word 文件的脚本。我将 Word 转换 PDF 的代码写在下方，供你参考。

复制代码

```
from win32com import client

def word2pdf(filepath, wordname, pdfname):
    worddir = filepath
    # 指定Word类型
    word = client.DispatchEx("Word.Application")
    # 使用Word软件打开a.doc
    file = word.Documents.Open(f"{worddir}\{wordname}", ReadOnly=1)
    # 文件另存为当前目录下的pdf文件
```

```
    file.ExportAsFixedFormat(f"{worddir}\{pdfname}", FileFormat=17,  Item=7, C
    # 关闭文件
    file.Close()
    # 结束word应用程序进程
    word.Quit()
```

我来为你详细解释一下上面这段代码。这段代码中我定义了一个 word2pdf() 函数，它被 Python 调用时，会根据自己的参数将 word 文件的路径、word 文件的名称和 pdf 名称传入参数中。

根据这些参数 word2pdf() 函数会调用 DispatchEx() 打开 Word 程序，再使用 Open() 函数打开 a.doc 文件，并使用 ExportAsFixedFormat() 函数将 Word 文件另存为 PDF 文件之后，使用 Close() 和 Quit() 关闭 a.doc 文件并结束 Word 进程。

由于 win32com 是用过 Word 的 VBA 脚本实现对 Word 进程行为的控制的，所以它的操作步骤和手动操作非常相似，所以这段代码也非常容易理解。

那在这里我需要提醒你注意两个容易被忽视的问题，第一个是由于 win32com 库在 Windows 系统上是直接调用 Word 进程实现格式转换的，因此你必须为当前的 Windows 安装 Word 以及 Office 的办公组件。那另一个是由于 win32com 对 Word 操作的方式是基于 Word 的 VBA 脚本，所以你想在转换时为 ExportAsFixedFormat() 函数增加新的参数，需要参考 Office 官方网站的文档。

我也将 Office 官方网站关于 Word 的 VBA 脚本所在网页提供给你做参考（https://docs.microsoft.com/zh-cn/office/vba/api/word.document.exportasfixedformat）。当你增加新的功能时，可以通过网页的内容来获得更多功能。

将 Excel 表格转换为 PDF

在你掌握了如何使用 win32com 实现 Word 文档转换为 PDF 之后，我再来带你实现 Excel 表格自动转换为 PDF 的功能，你也可以对比着来学习掌握，同时也注意观察把 Excel 和 Word 文件转换为 PDF 的主要差别。

Excel 表格默认保存的文件格式是 xls 或 xlsx，将 xls 或 xlsx 格式转换为 PDF 的步骤和思路与 Word 文档转换为 PDF 相同，所以我先把代码提供给你，然后再为你讲解。代码如

下：

复制代码

```
from win32com import client

def excel2pdf(filepath, excelname, pdfname):
    exceldir = filepath
    # 指定Excel类型
    excel = client.DispatchEx("Excel.Application")
    # 使用Excel软件打开a.xls
    file = excel.Workbooks.Open(f"{exceldir}\{excelname}", False)
    # 文件另存为当前目录下的pdf文件
    file.ExportAsFixedFormat(0, f"{excel}dir}\{pdfname}")
    # 关闭文件
    file.Close()
    # 结束excel应用程序进程
    excel.Quit()
```

对比 word2pdf() 和 excel2pdf()，你会发现实现的基本逻辑是相同的，但是在实现 Excel 转换上，有两个主要函数的参数不同。

1. DispatchEx() 函数的参数，这里使用了“Excel.Application”字符串作为该函数的参数，“Excel.Application”作为 DispatchEx() 参数的目的是，让 pywin32 库启动 Excel 进程，并让它读取“a.xls”文件。
2. ExportAsFixedFormat 函数的第一个参数从 pdf 路径变为保存的类型，你可以参考 Office 的 Excel VBA 官方文档，从中学习函数的每个参数。

将 PowerPoint 幻灯片转换为 PDF

在你学习了 Word 文档和 Excel 表格转换 PDF 文件的基础上，我再来带你你对比学习一下如何将 PowerPoint 的默认保存文件 ppt 格式转换为 PDF。参考 word2pdf() 和 excel2pdf() 两个函数，我直接将 PowerPoint 的幻灯片转换 PDF 文件的代码提供给你，你可以通过官方文档 (https://docs.microsoft.com/zh-cn/office/vba/api/powerpoint.presentation.exportasfixedformat)，来试着分析 ppt2pdf() 函数的参数和函数里的每行代码。

复制代码

```
from win32com import client

```

```
def ppt2pdf(filepath, pptname, pdfname):
    pptdir = filepath
    # 指定PPT类型
    ppt = client.DispatchEx("PowerPoint.Application")
    # 使用ppt软件打开a.ppt
    file = ppt.Presentations.Open(f"{pptdir}\{pptname}", False)
    # 文件另存为当前目录下的pdf文件
    file.ExportAsFixedFormat(f"{pptdir}\{pdfname}")
    # 关闭文件
    file.Close()
    # 结束excel应用程序进程
    excel.Quit()
```

显而易见，PowerPoint 幻灯片转换为 PDF 文件的代码逻辑和 word2pdf() 和 excel2pdf() 函数的逻辑完全相同，只有两处有所不同，一个是 DispatchEx() 的参数为“PowerPoint.Application”，它的功能是让 win32com 库打开 PowerPoint 进程。另一个是 ppt.Presentations.Open() 打开的对象不同，它打开的是每一页的 PPT，而 Excel 打开的是每个 sheet 对象。

以上就是如何将 Office 组件的 Word、Excel 与 PowerPoint 的默认保存文件格式转换为 PDF 格式。在转换之后，我们经常为了保护知识产权，或者增强文件的安全性，需要给 PDF 文件增加水印。所以接下来我就带你学习如何通过增加水印提高 PDF 文件的安全性。

提高 PDF 文件的安全性

安全性包括很多方面，比如为文档增加密码，从而增强保密性；也可以为文档增加水印，提升它的不可伪造性。如果你对前面课程熟悉的话，就能联想到我们在第 27 节课讲过，可以利用我们自动压缩的方式来给文件增加密码，所以我在今天这节课，主要以如何给 PDF 文件增加水印。

为 PDF 增加水印

为 PDF 文件增加水印，你可以通过 pyPDF2 库来实现。使用 pyPDF2 库的好处是，你不需要在当前计算机上安装任何 PDF 编辑器，就可以给 PDF 文件增加水印。

基于 pyPDF2 库来给 PDF 文件增加水印的原理是，把只带有水印的 PDF 文件，和需要增加水印的 PDF 文件合并即可。根据这一原理，你大概就能想到增加水印的步骤了。主要有三步，分别是：准备水印文件、安装 pyPDF2 库以及合并两个 PDF 文件。

老规矩，我们还是先从准备水印文件开始学习。带有水印的 PDF 文件，可以使用 Word 软件来生成。具体操作方法是：通过 Word 的“设计”-“水印”功能，来定制你自己的水印文件，接着再把它另存为 PDF 格式，之后这个带有水印的文件你就可以反复使用了。

接下来是安装 pyPDF2 的第三方库。由于它的安装包和软件同名，所以可以使用 pip 命令直接安装。安装后，我需要通过该库实现 PDF 文件的读写。我使用了这个库用于读写 PDF 文件的“PdfFileReader, PdfFileWriter”两个包，导入的代码如下：

复制代码

```
from PyPDF2 import PdfFileReader, PdfFileWriter

```

第三步，也是最重要的一步，**合并两个 PDF 文件**。合并文件需要使用 **pyPDF2 库的** mergePage() 函数 ** 实现。在实际工作中，我们通常需要给 PDF 文件的每一页都增加水印，此时我们需要使用 for 循环来迭代每一页，迭代之后，再把合并好的 PDF 文件保存为新的文件即可。

我把合并两个 PDF 文件的代码写在下面，然后再带你分析整个合并流程。

复制代码

```
from PyPDF2 import PdfFileReader, PdfFileWriter

def watermark(pdfWithoutWatermark, watermarkfile, pdfWithWatermark):

    # 准备合并后的文件对象
    pdfWriter = PdfFileWriter()

    # 打开水印文件
    with open(watermarkfile, 'rb') as f:
        watermarkpage = PdfFileReader(f, strict=False)

        # 打开需要增加水印的文件
        with open(pdfWithoutWatermark, 'rb') as f:
            pdf_file = PdfFileReader(f, strict=False)

            for i in range(pdf_file.getNumPages()):
                # 从第一页开始处理
                page = pdf_file.getPage(i)
                # 合并水印和当前页
                page.mergePage(watermarkpage.getPage(0))
                # 将合并后的PDF文件写入新的文件
```

```
22                 pdfWriter.addPage(page)
23
24     # 写入新的PDF文件
25     with open(pdfWithWatermark, "wb") as f:
26         pdfWriter.write(f)
```

在这段代码中，我定义了函数 watermark()，用来实现为 PDF 文件增加水印的功能。它的实现思路是先读取水印 PDF 文件，再把水印 PDF 文件与需要增加水印的文件的每一页进行合并。合并之后，我通过使用 PdfFileWriter() 类，产生了新的对象 pdfWriter，并将合并后产生的新 PDF 文件存储在 pdfWriter 对象中。最后，在所有页处理完成后，将合并后的 PDF 文件保存到“pdfWithWatermark”对象指向的文件中。

在这段代码中你需要注意的是，我使用了“with 方式”打开了文件，在文件处理完成前如果关闭文件的化，会出现“file close error”异常。因此你需要注意代码第 9、13 行的 with 缩进，而写入新的文件可以在水印 PDF 文件和要增加水印的文件关闭后进行，所以代码的第 25 行“with 语句”缩进可以在它上面的两个 with 代码块以外进行编写。

为了让你更直接地感知到增加水印后的结果，我把增加水印后的结果贴在下方，供你参考。

以上就是我使用 PyPDF2 库为 PDF 增加水印的全部流程。不过除了增加水印外，你还能使用 pdfWriter 对象，来实现很多实用的功能，比如为 PDF 文件设置密码、插入新的页面、旋转 PDF 页面等等。

此外，由于 pyPDF2 库封装得非常好，所以它的调用很简单，你只需一个函数就能实现我刚才提到的这些功能了。我将 pyPDF2 库的官方文档链接（https://pythonhosted.org/PyPDF2/）贴在这里，当你需要操作 PDF 文件实现其他功能时，可以参考官方文档中 PdfFileWriter() 函数的参数，为不同的功能增加相应参数即可。

小结

最后，我来为你总结一下今天的主要内容。在本讲中，我为你讲解了如何通过 pywin32 库把 Offce 组件常用的 doc、docx、xls、xlsx、ppt、pptx 等文件转换为 PDF 文件格式的

通用方法。这个通用的方法就是**通过 pywin32 库的 COM 编程接口，调用 VBA 脚本实现格式的转换。**

你学会了 pywin32 库之后，除了能把这些办公文件转换为 PDF 文件外，还能对 Office 组件中的任意一个软件进行**常见功能的调用**，因为 **pywin32 调用的 VBA 脚本和 Office 宏的脚本是完全相同的。

我在本讲中除了为你讲解了 pywin32 库外，还讲解了 pyPDF2 库。pyPDF2 库能够在你将文件转换为 PDF 之后，还能对 PDF 的格式和内容进行微调，让你的 PDF 文件批量处理能达到手动处理的文件精细程度。

最后，你还可以把 PDF 文件和上一讲中的自动收发邮件，以及我们学习过的 Word 自动处理相结合，把 PDF 格式的合同作为邮件附件进行文件的自动生成和邮件的自动发送。

思考题

按照惯例，我来为你留一道思考题。如果在一个文件夹中既有 Word 文件，又有 PowerPoint 文件，你该如何将文件夹中的这些类型的文件，批量转换为 PDF 文件呢?

提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 29 | 网络和邮件库：定时收发邮件，减少手动操作

下一篇 结课测试题 | 这些Python自动化办公的知识你都掌握了吗？

精选留言 (2)

写留言

innovator琳
2021-04-25

Mac os 上怎么操作?

展开

作者回复: 在Mac 系统上就没有这么便利了， 像是word转换为pdf要用 docx2pdf， excel最方法的是pandas ， 而Powerpoint最方便且格式上处理的比较好的是wps office

1

LDxy
2021-04-20

COM编程接口是一种接口类型吗?

展开

作者回复: 首先，你可以将COM理解为DLL提供的一系列的可以操纵Windows的函数；其次还要把COM当作规范，因为你可以跨语言来使用它提供的函数，本讲中的Word、Excel、Powerpoint使用到的另存为函数，就是COM编程接口提供的，更详细的解释可以参考：https://baike.baidu.com/item/com%E7%BC%96%E7%A8%8B